Year 4 — Numerical Linear Algebra

Based on lectures by Prof. Yuji Nakatsukasa

Notes taken by James Arthur

Michaelmas 2022

Contents

1 Introduction

The goal of this course is to solve and understand problems using (usually non-square) matrices. The course will cover,

1. SVD. Singular Value Decomposition, see Linear Algebra (Year 2). $A = U\Sigma V^T$.
2. Linear systems $Ax = b$ and eigenvalue problems $Ax = \lambda x$. Least square problems, $\min_x \|Ax - b\|_2$.

There are going to be three classes of approaches,

- Direct Methods, classical approach. Not useful for very big matrices.
- Iterative solvers, work even if matrix sizes are larger.
- Randomised Algorithms, use some sense of randomisation in order to get an algorithm that works with high probability better than direct for very large matrices.

1.1 Why do we care?

When we want to solve a non-linear equation a linear system pops out. Motivation: minimising a function, $\min_x f(x)$ where $f : \mathbb{R}^n \to \mathbb{R}$, where we let n be large. One powerful way to find a minimum is to find a stationary point. So we find,

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \dots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} = 0$$

if we have a nice (convex) function, then if we have a stationary point, we have a minimiser. This boils down to, letting $F = \nabla f : \mathbb{R}^n \to \mathbb{R}^n$. We need to find zeros of this non-linear problem. Hence we now have a root-finding problem, and so we can use Newton's Method.

$$x_{\text{new}} = x_{\text{old}} - \mathcal{J}^{-1} F(x_{\text{old}})$$

and we see,

$$\mathcal{J}_{ij} = \frac{\partial F_i}{\partial x_j}$$

and then this is the hessian of f. This is then a linear system,

$$\mathcal{J}\Delta x = F(x_{\text{old}}).$$

NB! Here $A^* = \overline{A}$ (instead of $A^* = \overline{A}^T$) We are going to stay under real matrices because there are only two cases where the difference matters. Further, $m \geq n$ for most matrices in this course. For orthonormal matrices,

$$A^T A = I_n \qquad AA^T \neq I_m$$

1.2 Norms

We need norms to quantify how big matrices are. These help us when approximating matrices. Given some $\mathbf{x} \in \mathbb{R}^n$, we have

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

and 1-norm,

$$\|x\|_1 = |x_1| + \cdots + |x_n|$$

and the p-norm,

$$\|x\|_p = (|x_1|^p + \cdots + |x_p|^p)^{\frac{1}{p}}$$

and finally the ∞-norm,

$$\|x\|_\infty = \max_i |x_i|$$

Definition 1.1 (Norm)**.** A norm satisfies three axioms,

- $\|\alpha x\| = |\alpha| \, \|x\|$
- $\|x\| \geq 0$ and $\|x\| = 0 \iff x = 0$
- $\|x + y\| \leq \|x\| + \|y\|$

Lemma 1.2 (Holder's Inequality)**.** For $p > q$,

$$\|x\|_p \leq \|x\|_q$$

Definition 1.3 (Unitarily Invariant)**.** If A is orthonormal, then $\|Ax\|_2 = \|x\|_2$.

Lemma 1.4 (Cauchy-Schwartz)**.** For any $x, y \in \mathbb{R}^n$,

$$|x^T y| \leq \|x\|_2 \, \|x\|_2$$

Proof. For any scalar c, $\|x - cy\|_2^2 = \|x\|_2^2 - 2cc^T y + \|y\|_2^2$. Now we complete the square and we get,

$$\begin{aligned} \|x - cy\|_2^2 &= \|x\|_2^2 - 2cc^T y + \|y\|_2^2 \\ &= \|y\|_2^2 \left(c - \frac{x^T y}{\|y\|_2} \right)^2 + \|x\|_2^2 - \frac{(x^T y)^2}{\|y\|_2^2} \end{aligned}$$

and so we now minimise by letting $c = \frac{x^T y}{\|y\|_2}$ and so we get Cauchy Schwartz. □

Now for matrix norms. We have the p-norm,

$$\|A\|_p = \sup_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p} = \max_{\|x\|_p = 1} \frac{\|Ax\|_p}{\|x\|_p}.$$

The most important case is when $p = 2$, this is the spectrum norm. This is,

$$\|A\|_2 = \max_{\|x\|_2 = 1} \|Ax\|_2$$

Exercise. Show,

$$\|A\|_1 = \text{maximum column sum}$$
$$\|A\|_\infty = \text{maximum row sum}$$

Definition 1.5 (Frobenius Norm)**.**

$$\|A\|_F = \sqrt{\sum_i \sum_j |A_{ij}|^2} = \sqrt{\mathbf{A}^T \mathbf{A}} = \sqrt{\text{tr}(A^T A)}$$

where

$$\mathbf{A} = \begin{pmatrix} A_{11} \\ \vdots \\ A_{1n} \\ A_{21} \dots \end{pmatrix}$$

Definition 1.6 (Trace Norm)**.**

$$\|A\|_* = \sum_{i=1}^{\min(m,n)} \sigma_i(A)$$

For most p-norms,

$$\|AB\|_p \leq \|A\|_p \|B\|_p$$

and for the Frobenius norm,

$$\begin{aligned}\|AB\|_F &\leq \|A\|_F \|B\|_F \\ &\leq \|A\|_2 \|B\|_F\end{aligned}$$

this is the subordinate property. Where this goes wrong is,

$$\|A\|_\infty = \max_{i,j} |A_{ij}|$$

Here is a result on subspaces,

Lemma 1.7. $\mathcal{S}_1 = \text{span}(V_1)$ and $\mathcal{S}_2 = \text{span}(V_2)$ where $V_1 \in \mathbb{R}^{n \times d_1}$ and $V_2 \in \mathbb{R}^{n \times d_2}$, with $d_1 + d_2 > n$. Then $\exists x \in 0 \in \mathcal{S}_1 \cap \mathcal{S}_2$.

2 SVD

We already know of the symmetric eigenvalue decomposition $A = V\Lambda^T V$ for a symmetric $A \in \mathbb{R}^{n\times n}$ where $V^T V = I_n$ and $\Lambda = \text{diag}(\lambda_1, \dots \lambda_n)$. Now we learn about the Singular Value Decomposition (SVD). This is for any $A \in \mathbb{R}^{m\times n}$ for $m \geq n$. Here $U^T U = V^T V = I_n$, $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n)$ where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. We now want to prove this always exists,

Proof. Take some $A^T A$ (Gram Matrix) symmetric positive semi-definite. That is, all the eigenvalues are nonnegative. We then have an eigenvalue decomposition,

$$A^T A = V \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} V^T$$

Now we let $B = AV$. Then,

$$\begin{aligned} B^T B &= V^T A^T A V \\ &= \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} = \Sigma^2 \end{aligned}$$

Suppose $\lambda_n > 0$. Then let,

$$U = B \begin{pmatrix} \lambda_1^{\frac{1}{2}} & & \\ & \ddots & \\ & & \lambda_n^{\frac{1}{2}} \end{pmatrix}$$

and $U^T U = \Sigma^{-1} B^T B \Sigma^{-1} = I_n$. Now $A = BV^T$ and $B = U\Sigma$ and so $A = U\Sigma V^T$. Now consider when $\lambda_{r+1} = 0$. We have,

$$\Sigma = \begin{pmatrix} \lambda_1 & & & & & \\ & \ddots & & & & \\ & & \lambda_r & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix}$$

and now we can do something sensible. We do,

$$\begin{pmatrix} U_r & 0 \end{pmatrix} = B \begin{pmatrix} \lambda_1^{\frac{1}{2}} & & & & & \\ & \ddots & & & & \\ & & \lambda_r^{\frac{1}{2}} & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{pmatrix}$$

instead of 0's. We still have $A = BV^T$, but,

$$U = BV^T = \begin{pmatrix} U_r & 0 \end{pmatrix} \begin{pmatrix} \Sigma_r & \\ & I \end{pmatrix} \begin{pmatrix} V_r^T \\ V_\perp^T \end{pmatrix} = U\Sigma_r V^T$$

This is the economical SVD. □

We also have the full SVD where, $A = \begin{pmatrix} U & U_\perp \end{pmatrix} \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T$ where $U \in \mathbb{R}^{m \times m}$ orthogonal.

From the SVD we get,

- rank r of $A \in \mathbb{R}^{m \times n}$, number of nonzero singular values $\sigma_i(A)$. We can always write $A = \sum_{i=1}^{\text{rank}(A)} \sigma_i u_i v_i^T$,
- Column Space, span of $U = [u_1, \ldots, u_r]$,
- row spane, row span of $v_1^T, \ldots, v_r^T$,
- null space, $v_{r+1}, \ldots, v_n$.

We note that the SVD can be written in the form of an outer product, $A = U\Sigma V^T = \sum_{i=1}^{k} \sigma_i U_i V_i^T$. We also note that $Av_i = \sigma_i u_i$. Let's prove this,

$$\begin{aligned} Av_i &= \left(\sum_{j=0}^{n} \sigma_j u_j v_i^T \right) v_i \\ &= \sigma_i u_i \end{aligned}$$

and similarly, $u_i^T A = \sigma_i v_i^T$. Further we can show if $A = U\Sigma V^T$ we can say,

$$A^T A = V\Sigma^2 V^T \tag{1}$$

or

$$AA^T = U\Sigma^2 U^T \tag{2}$$

Further for any 1 such that $A = \tilde{U}\Sigma V^T$ and $A = U\Sigma\tilde{V}^T$ then **some result**.
We call a triplet a singular triple if $Av_i = \sigma_i u_i$ and $u_i^T A = \sigma_i v_i^T$ both hold. We now ask whether the SVD is unique. We can see,

$$\begin{aligned} A &= U\Sigma V^T \\ &= USS^{-1}\Sigma SS^{-1}V^T \\ &= (US)(S\Sigma S)(SV^T) \end{aligned}$$

where we just let S be diagonal with ± 1. We see that the grouped terms retain the structure that we need to have a new SVD. Further this SVD is different, but Σ is the same ($S = S^{-1}$ and Σ is diagonal). When $\sigma_i = \sigma_j$, we have larger degree og freedom, i.e. $\sigma_1 = \sigma_2$. We can write the first two diagonals as $\sigma_1 QQ^T$. We then have,

$$A = U \begin{pmatrix} Q & \\ & I_{n-2} \end{pmatrix} \Sigma \begin{pmatrix} Q^T & \\ & I_{n-2} \end{pmatrix} V^T$$

If A is orthogonal, we can say the following are SVD's of A,

$$A = AII = IIA = (AQ)IQ^T.$$

Lemma 2.1.

$$\|A\|_2 = \sigma_1(A)$$

Proof. Use SVD,

$$\begin{aligned} \|Ax\|_2 &= \left\|U\Sigma V^T x\right\|_2 \\ &= \left\|\Sigma V^T x\right\| \\ &= \|\Sigma y\|_2 \end{aligned}$$

$$= \sqrt{\sum_{i=1}^{n} \sigma_i^2 y_i^2}$$
$$\leq \sqrt{\sum_{i=1}^{n} \sigma_1^2 y_i^2} = \sigma_1 \|y\|_2^2 = \sigma_1$$

□

2.1 Applications

2.1.1 Low-rank approximations

Given some $A \in \mathbb{R}^{m\times n}$, we want to find some sort of A_r such that $A =\approx A_r = U_r\Sigma_r V_r^T$. You may ask why?

- Let A be stupidly large, then this saves storage.
- Matrix multiplication is of order $\mathcal{O}(mn)$, but in terms of this approximation we have something of the order of $\mathcal{O}((m+n)r)$, which is a massive saving.

Low-rank Matrices When talking about low-rank matrices, we say some $\text{rank}(B) \leq r$. If this is true we can write it as, $B = xy^T$. Let's prove this,

Proof. We know $B = U\Sigma V^T$, then we can truncate $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r)$ and then let $U_r\Sigma = X$ and $Y = V$. Conversely, if $B = XY^T$, then $B = U_X\Sigma_X V_X^T$ then $Y = U_Y\Sigma_Y V_Y^T$, then we look at $XY^T = U_X\Sigma)XV_X^T V_Y\Sigma_Y U_Y^T$. Now we SVD AGAIN!!!!! $V = U_X\tilde{U}\tilde{\Sigma}\tilde{V}^T U_Y^T$ AND AGAIN! $\tilde{\tilde{U}}\tilde{\tilde{\Sigma}}\tilde{}^T\tilde{V}$. Then we can say that V can have at most r positive singular values. □

Now we want to find B such that $\text{rank}(B) \leq r$. So we want to minimise $\|A - B\|_2$ given A. We see $\|A - B\|_2 \leq \|A - C\|_2$ for all $\text{rank}\, C \leq r$. That is, we want to solve,

$$A = U\Sigma V^T = \sum_{n=1}^{n} s_i u_i v_i$$

and then truncating B,

$$B = U_r\Sigma_r V_r^T = \sum_{i=0}^{r} \sigma_i u_i v_i^T$$

We take the truncated SVD, $A_r = U_r\Sigma_r V_r^T$ where $\Sigma_r = \text{diag}(\sigma_1, \ldots, \sigma_n)$.

Lemma 2.2.

$$\|A - A_r\| = \sigma_{r+1} = \min_{\text{rank}\, B - r} \|A - B\|_2$$

Proof. Since $\text{rank}\, B \leq r$, then we can write $B = B_1 B_2^T$ where B_1, B_2 have r columns. There is some orthonormal $W \in \mathbb{C}^{n\times(n-r)}$ such that $BW = 0$. Then we can say,

$$\begin{aligned}\|A - B\|_2 &\geq \|(A - B)W\|_2 \\ &= \|AW\|_2 \\ &= \left\|U\Sigma(V^T W)\right\|_2\end{aligned}$$

Now since W is $(n-r)$ dimensional, there is an intersection between W and $[v_1, \ldots v_{r+1}]$, the $(r+1)$-dimensional subspace spanned by the leading $r+1$ left singular values, that is $[W, v_1, \ldots, v_{r-1}]\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0$ has a solution, then Wx_1 is such a vector. Then we scale x_1, x_2 to have unit norm, and $\left\|U\Sigma V^T W x_1\right\|_2 = \|U_{r+1}\Sigma_{r+1}x_2\|$ where U_{r+1}, Σ_{r+1} are leading $r+1$ parts of U, Σ. □

2.2 Courant-Fischer minmax Theorem

We say that the λ_i of a symmetric or hermitian Λ matrix is,

$$\lambda_i(\Lambda) = \max_{\dim \mathcal{S}=i} \min_{x \in \mathcal{S}} \frac{x^T A x}{x^T x}$$

and analogously for an rectangular $A \in \mathbb{C}^{m \times n}$, we have,

$$\sigma_i(A) = \max_{\dim \mathcal{S}=i} \min_{x \in \mathcal{S}} \frac{\|Ax\|_2}{\|x\|_2}$$

One helpful way to look at this is maybe,

$$\min_{x \in \mathcal{S}, \|x\|_2 = 1} \|Ax\|_2 = \min_{Q^T Q = 1, \|y\|_2 = 1} \|AQy\|_2 = \sigma_{\min}(AQ) = \sigma_i(AQ)$$

Corollary 2.3. For the singular values for any matrix A,

- $\sigma_i(A + E) \in \sigma_i(A) + [-\|E\|_2, \|E\|_2]$
- Special Case, $\|A\|_2 - \|E\|_2 \leq \|A - E\|_2 \leq \|A\|_2 + \|E\|_2$

We can say something similar for eigenvalues of a symmetric matrix.

Here are some more applications of Courant-Fischer. Consider some matrix, $\begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$, then we can say,

$$\sigma_i\left(\begin{pmatrix} A_1 \\ A_2 \end{pmatrix}\right) \geq \max(\sigma_i(A_1), \sigma_i(A_2))$$

Proof.

$$\begin{aligned} \sigma_i(A) &= \max_{\dim S = i} \min_{x \in S, \|x\|_2 = 1} \left\| \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} x \right\| \\ &\leq \max_{\dim S = i} \min_{x \in S, \|x\|_2 = 1} \|A_1 x\|_2 = \sigma_i(A_1) \end{aligned}$$

□

3 $\mathbf{Ax} = \mathbf{b}$

What if $A_{11} = 0$? For example $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. We should be able to do this! Solution, pivot. This reorders the rows of A_i. So instead of,

$$A_i = \ell_i u_i^T + \begin{pmatrix} & 0 & \\ 0 & * & * \\ & * & * \end{pmatrix}$$

we aim for,

$$P_1 A = \ell_i u_i^T + \begin{pmatrix} & 0 & \\ 0 & * & * \\ & * & * \end{pmatrix}$$

where P_i is a permutation matrix. For example, $P_i = \begin{pmatrix} 0 & & & 1 \\ & & 1 & 0 \\ & \iddots & & \\ 1 & & & \end{pmatrix}$ this says that $P_i A$ exchanges the rows and AP_i exchanges columns. We are going to use these P_i's to just move the largest non-zero entries to the top. So instead of what we expect to have, we have,

$$\tilde{P}_1 P_0 A = \begin{pmatrix} 1 & 0 \\ 0 & P_1 \end{pmatrix} \left(\ell_1 u_1^T + \begin{pmatrix} & 0 \\ 0 & A_1 \end{pmatrix} \right)$$

where, $\tilde{P}_1 = \begin{pmatrix} 1 & 0 \\ 0 & P_1 \end{pmatrix}$, and so we redefine, $\ell_1 = \tilde{P}_{n-1} \dots \tilde{P}_0 \ell_1$ and so we now have,

$$\tilde{P}_{n-1} \dots \tilde{P}_0 = \tilde{\ell}_1 u_1^T + \dots \tilde{\ell}_n u_n^T$$

but does this hurt our structure? Unfortunately not. This is because each $\tilde{P}_n$ only acts on the structure of the part we want it to. Hence we have we LU factoriation,

$$A = \begin{pmatrix} 0 & & & \\ \ell_1 & 0 & & \\ \vdots & \ell_2 & 0 & \\ & \vdots & \ddots & \\ & & & \ell_n \end{pmatrix}$$

3.1 Cholesky Factorisation

We have $A = A^T \succ 0$, we have $\text{eigs}(A) > 0$. We are going to force $L = U$ throughout the operation. Hence,

$$A = \ell_1 \ell_1^T + \begin{pmatrix} & 0 \\ 0 & A_1 \end{pmatrix}$$

We can say that $A_{11} > 0$ and that, $A_1 \succ 0$. Then RECURSION! Hence we have, after n steps, we have the Cholesky, $R^T R$. We recall the $A^T A$ Gram Matrix. If we have some indefinite matrix, when $A = A^*$ we have $A = LDL^*$ where L is triangular, but D is a block matrix.

3.2 QR Factoriation

Consider some over-determined problem, $Ax = b$ usually not possible. Then we want to $\min_x \|Ax - b\|_2$. To solve, $A = LU$ is useless. We need something different, we need the QR factorisation. We let $A = QR$, where $Q^T Q = I$ and R is upper triangular. This exists for any A.

3.3 Householder QR

In order to do Householder QR, we need to look at Householder Reflectors, a useful tool. These are a set of orthogonal matrices such that $H = I_m - 2vv^T$, where $v = (v_1, \dots, v_m)^T$ and $\|v\|_2 = 1$. We know these matrcies are symmetric, orthogonal. Let's prove this,

Proof. We see,

$$\begin{aligned} H^T H = H^2 &= (I - 2vv^T)(I - vv^T) \\ &= I - 4vv^T + 4vv^T vv^T = I \end{aligned}$$

□

and we have an eigenvalue decomposition, $H = I - 2vv^T$, which is,

$$\begin{aligned} H &= I - 2vv^T \\ &= \begin{pmatrix} V & V_\perp \end{pmatrix} \begin{pmatrix} 1 & & \\ & \ddots & \end{pmatrix} \begin{pmatrix} v^T \\ v_\perp^T \end{pmatrix} - \begin{pmatrix} V & V_\perp \end{pmatrix} \begin{pmatrix} 2 & & \\ & 0 & \\ & & \ddots \end{pmatrix} \begin{pmatrix} v^T \\ v_\perp^T \end{pmatrix} \\ &= \begin{pmatrix} V & V_\perp \end{pmatrix} \begin{pmatrix} -1 & & & \\ & 1 & \ddots & \\ & & & 1 \end{pmatrix} \begin{pmatrix} v^T \\ v_\perp^T \end{pmatrix} \end{aligned}$$

Then we have a lemma,

Lemma 3.1. There is some $H = I - 2vv^T$ such that $Hu = w$ and $Hw = u$, where $\|u\|_2 = \|w\|_2$.

Proof. Consider $v = \frac{u-w}{\|u-w\|_2}$. Then,

$$\begin{aligned} Hu &= (I - 2vv^T)u \\ &= u - 2v(v^T u) \\ &= u - 2(u-w)\frac{(u-w)^T u}{\|u-w\|_2} \\ &= u - 2(u-w)\frac{\|u\|^2 - w^T u}{2\|u\|^2 - 2w^T u} \\ &= w \end{aligned}$$

For the other, we see,

$$Hw = H(Hu) = Iu = u.$$

□

This is useful in the very specific case where, $w = (\|w\|_2, 0, \dots, 0)^T$.

3.3.1 Householder QR

We now want to consider some $H_1 A$, then our first column is just going to zero apart from the top which is $\|w\|_2$. Now we get another householder reflector, but we need to be careful not to break the work we have done already. Hence we consider $H_2 H_1 A$, but $H_2 = I - 2v_2 v_2^T$, but $v_2 = (0, *, \dots, *)$, this helps us keep the structure of this matrix. We carry on to get, $H_n \dots H_1 A$ where they all have v_k's with the first $k-1$ entries zero. Then we get some,

$$A = Q_F \begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} Q & Q_\perp \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} = QR,$$

where this is the full QR factorisation. Consider applying Householder QR to A, which is orthonormal. Then $A^TA = I$, then,

$$A = Q_F \begin{pmatrix} R \\ 0 \end{pmatrix},$$

but as A is orthonormal. We can write,

$$A = \begin{pmatrix} A & Q_\perp \end{pmatrix} \begin{pmatrix} I \\ 0 \end{pmatrix}$$

and hence we have the proof of existence of orthogonal complement. This is a neat constructive proof.

3.3.2 Least Squares using QR

Consider some $\|Ax - b\|_2$, where we can't solve this exactly, so it can't be 0. We want to minimise this.

$$\begin{aligned} \|Ax - b\| &= \left\| Q_F \begin{pmatrix} R \\ 0 \end{pmatrix} x - b \right\|_2 \\ &= \left\| \begin{pmatrix} R \\ 0 \end{pmatrix} x - Q_F^T b \right\|_2 \\ &= \left\| \begin{pmatrix} R \\ 0 \end{pmatrix} x - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right\|_2 \\ &= \left\| \begin{pmatrix} Rx - b_1 \\ -b_2 \end{pmatrix} \right\| \end{aligned}$$

Hence to minimise, we want $Rx = b_1$, hence $x = R^{-1}Q^Tb$. Hence we see no Q_F or $Q_\perp$ is needed. Hence, we compute the QR factorisation, then $x = R^{-1}Q^Tb$.

It turns out we can also rewrite it in terms of the normal equation,

$$A^TAx = A^Tb,$$

this is an unstable algorithm, so even if it looks simpler, we prefer not to use it.

3.4 Numerical Stability

When we consider $Ax = b$, we plug this into a computer and then get that it outputs some $\hat{x}$ such that $A\hat{x} = b$, this hows a massive flaw, we know that $Ax = b$ and $A\hat{x} \approx b$, but this doesn't mean that $x \approx \hat{x}$, these can be quite different. We have two ideas, conditioning and instability. Conditioning is a property of the problem, and instability is a property of an algorithm, there aren't interchangable.

The conditioning is like the sensitivity of the problem under perturbation. Given the task of computing $Y = f(X)$ (e.g. $X = (A, b)$ and $Y = x$). Then we have,

$$\kappa = \sup \frac{\|f(x + \delta x) - f(x)\|}{\|\delta x\|}.$$

Further, we have κ_r, the relative conditioning number,

$$\kappa = \lim \|\delta x\| \to 0 \sup \frac{\|f(x + \delta x) - f(x)\|}{\|\delta x\|},$$

this does look very similar to a derivative. If $\kappa \gg 1$, then it's ill-conditioned. If $\kappa = \mathcal{O}(1)$, then it is well conditioned.

Definition 3.2 (Backward Stable)**.** An algorithm is called backward stable if the computed output $\hat{Y} = \text{fl(f(X))}$ can be written as $f(X + \delta X)$ where δX is small, that is, $\frac{\|\delta X\|}{\|X\|} = \mathcal{O}(\varepsilon)$ (machine precision).

Note. We note that backward stability doesn't mean accurate, but backward stability and f being well conditioned means it's accurate.

Rough Argument.

$$\begin{aligned}\left\|\hat{Y} - Y\right\| &= \|f(X + \delta X)\| \\ &\lesssim \kappa \|\delta X\| \\ &= \mathcal{O}(\|\delta X\|) = \mathcal{O}(\varepsilon \|X\|)\end{aligned}$$

□

Some examples of well conditioned problems are,

- Singular values $\kappa = 1$, $\sigma_i(A + E) \in \sigma_i(A) + [\|E\|]$,
- Eigenvalues of symmetric matrices, $\kappa = 1$,
- $Ax = b$ if A is well-conditioned.

We also note that for non-symmetric matrices can be ill-conditioned.

Now lets consider κ for $Ax = b$. We want to understand what the solution will look like if we peturb the input. Suppose $(A + \Delta A)\hat{x} = b + \Delta b$, but let $\Delta b = 0$ for simplicity. We know $\frac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\varepsilon)$. We have,

$$\begin{aligned}\hat{x} &= (A + \Delta A)^{-1} b \\ &= (A(I + A^{-1}\Delta A))^{-1} b \\ &= (I + A^{-1}\Delta A)^{-1} A^{-1} b \\ &= (I - A^{-1}\Delta A + (A^{-1}\Delta A)^2 + \mathcal{O}(\Delta A^3)) A^{-1} b \\ &= x - A^{-1}\Delta A^{-1} b + \mathcal{O}(\Delta A^3)\end{aligned}$$

Hence,

$$\begin{aligned}\frac{\|x - \hat{x}\|}{\|x\|} &= \frac{\left\|A^{-1}\Delta A x\right\|}{\|x\|} \\ &\leq \frac{\left\|A^{-1}\Delta A\right\| \|x\|}{\|x\|} \\ &\leq \left\|A^{-1}\right\| \varepsilon \|A\| \\ &= \varepsilon \|A\| \left\|A^{-1}\right\|\end{aligned}$$

and so we get $\kappa_2(A) = \left\|A^{-1}\right\| \|A\|$, which is the condition number of A. In terms of the singular values,

$$\kappa_2 = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)} \geq 1.$$

4 Eigenvalue Problems

We consider $Ax = \lambda x$. Given A we want to find $\lambda \in \mathbb{C}$ and $x \in \mathbb{C}^n$. It is impossible to solve exactly if $n \geq 5$. This is just Galois Theory. We can show how we can go from a polynomial to a matrix. We fiddle with companion matrix,

Proof. If $p(\lambda) = 0$ then it is an eigenvalue of the following companion matrix,

$$C = \begin{pmatrix} -a_{n-1} & -a_{n-2} & \dots & -a_1 & -a_0 \\ 1 & & & & \\ & 1 & & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}$$

Then we see,

$$\begin{pmatrix} -a_{n-1} & -a_{n-2} & \dots & -a_1 & -a_0 \\ 1 & & & & \\ & 1 & & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} \lambda^{n-1} \\ \vdots \\ \lambda^2 \\ \lambda \\ 1 \end{pmatrix} = \begin{pmatrix} \lambda^n \\ \vdots \\ \lambda^3 \\ \lambda^2 \\ \lambda \end{pmatrix} = \lambda \begin{pmatrix} \lambda^{n-1} \\ \vdots \\ \lambda^2 \\ \lambda \\ 1 \end{pmatrix}.$$

□

We note that $\det(\lambda I - A)$ isn't feasible computationally. This may seem bad, but there is some good news. We are going to try to use orthogonal transformations as much as possible. The ultimate goal is to use the Schur form, $A = UTU^*$ where U is unitary and T is upper triangular. Given $\text{eigs}(A) = \text{eigs}(XAX^{-1})$ by a similarity transformation and the Schur transformation is a similarity transformation, and so $\text{eigs}(A) = \text{eigs}(UTU^*) = \text{eigs}(T)$ and the eigenvalues of T is just the diagonal entries. We further note that if A is normal, then T is diagonal. So, the Schur form is diagonal, $A = UDU^*$. Now we prove that these always exist,

Proof. Recall $\det(\lambda I - A) = 0$ and by the FTA then there is some $(\lambda_1 I - A)v = 0$ where $v \neq 0$. We consider $Av = \lambda_1 v$ and take $\begin{pmatrix} v & v_\perp \end{pmatrix}$ unitary. Then,

$$A \begin{pmatrix} v & v_\perp \end{pmatrix} = \begin{pmatrix} \lambda v & * \end{pmatrix}$$

and we can now see,

$$\begin{pmatrix} v v_\perp \end{pmatrix}^* A \begin{pmatrix} v & v_\perp \end{pmatrix} = \begin{pmatrix} \lambda_1 & * & * \\ 0 & & \\ \vdots & & A_2 \\ 0 & & \end{pmatrix}$$

Now we let $A_2 v_2 = \lambda_2 v_2$ where $V = \begin{pmatrix} v_2 & v_{2\perp} \end{pmatrix}$ and so we can now write,

$$\begin{pmatrix} 1 & 0 \\ 0 & V_2^* \end{pmatrix} U_1^* A U_1 \begin{pmatrix} 1 & 0 \\ 0 & V_2 \end{pmatrix} = \begin{pmatrix} \lambda_1 & & & * \\ 0 & \lambda_2 & & \\ \vdots & 0 & A_3 & \\ 0 & \vdots & & \\ 0 & 0 & & \end{pmatrix}$$

and so we carry on to get,

$$U_m^* \dots U_1^* A U_1 U_2 \dots U_m = T$$

□

4.1 Power Method

The power method computes on eigenpair, that is (λ, v) such that $Av = \lambda v$. The algorithm is very simple.

Let x be an arbitrary vector. Then let $x_{i+1} := \frac{Ax_i}{\|Ax_i\|_2}$ and repeat for k steps.

Then we have,

$$x_k = \frac{A^k x_0}{\|A^k x_0\|}$$

and we claim that if $\lambda_k = x_k^* A x_k$, then (λ_k, x_k) converge to an eigenpair, (λ_1, v_1). Further this is the dominant eigenpair. To prove this makes sense,

Proof. Assume A is diagonalisable. Then $A = VDV^{-1}$. Then $x_0 = Vc = \sum_{i=1}^n V_i c_i$ where $c = V^{-1}x$. Now assume $c_n \neq 0$. Now,

$$\begin{aligned}
x_k &= \frac{A^k x_0}{\|A^k x_0\|_2} \\
&= A^{k-1} \frac{Ax_0}{\|A^k x_0\|_2} \\
&= \frac{\sum c_i \lambda_i^k v_i}{\|A^k x_0\|_2} \\
&= \frac{1}{\|Ak^k x_0\|} \left(\lambda_1^k c_1 v_1 + \lambda_2^k c_2 v_2 + \cdots + \lambda_n^k c_n v_n\right) \\
&= C\left(v_1 + \frac{c_2}{c_1}\left(\frac{\lambda_2}{\lambda_1}\right)^k v_2 + \cdots + \frac{c_j}{c_1}\left(\frac{\lambda_j}{\lambda_1}\right)^k v_j + \dots\right).
\end{aligned}$$

Hence $x_k \to \pm v_1$ and so $x_k^* A x_k \to v_1^* \lambda_1 v_1 = \lambda_1$. □

5 QR Algorithm

This algorithm calculates all of the eigenvalues. Basically we are going to find the Schur Form, $A = UTU^*$. The algorithm is simple, we are going to say $A_1 = Q_1R_1$ and then $A_2 := R_1Q_1 = Q_2R_2$ and then, $A_3 := R_2Q_2 = Q_3R_3$ and the surprise is that A_k converges to upper triangular under mild assumptions. Further A_k is similar to A.

Theorem 5.1. We can say that, $A_{k+1} = Q^{(k)T}AQ^{(k)}$ and, $A^k = (Q_1 \dots Q_k)(R_k \dots R_1) = Q^{(k)}R^{(k)}$, where the we notate $Q^{(k)} := Q_1 \dots Q_k$.

Proof. We say,

$$\begin{aligned} A_{k+1} = R_kQ_k & \\ &= Q_k^TQ_kR_kQ_k \\ &= Q_k^TA_kQ_k \end{aligned}$$

and continue this many times to get,

$$A_{k+1} = Q_k^T \dots Q_1^TAQ_1 \dots Q_k := (Q^{(k)})^TAQ^{(k)}.$$

For the second part we prove by induction. Suppose the statement is true for $A^{k-1} = Q^{(k-1)}R^{(k-1)}$. We see,

$$\begin{aligned} Q^{(k-1)T}AQ^{(k-1)} = A_k &= Q_kR_k \\ A &= Q^{(k-1)}Q_kR_k(Q^{(k-1)})^T \\ A^k &= AA^{k-1} \\ &= Q^{(k-1)}Q_kR_kQ^{(k-1)T}Q^{(k-1)}R^{(k-1)} \\ &= Q^{(k)}R_kR^{(k-1)} = Q^{(k)}R^{(k)}. \end{aligned}$$

□

Understanding this, we link this it power method. We know $A^k = Q^{(k)}R^{(k)}$. Then consider,

$$\begin{aligned} A\begin{pmatrix}1\\0\\0\\\vdots\\0\end{pmatrix} &= Q^{(k)}R^{(k)}\begin{pmatrix}1\\0\\0\\\vdots\\0\end{pmatrix} \\ &= \begin{pmatrix}R_{11}\\0\\0\\\vdots\\0\end{pmatrix} \\ &= Cq_1^{(k)} \\ &\to v_1 \end{aligned}$$

Then we can say,

$$A_{k+1} = Q^{(k)^T}AQ^{(k)} = \begin{pmatrix}\lambda_1 & * \\ 0 & \\ \vdots & \\ 0 & \end{pmatrix}.$$

We have better news. Start with $A^k = Q^{(k)}R^{(k)}$, and then invert, $A^{-k} = (R^{(k)})^{-1}(Q^{(k)})^T$ and then transpose, $A^{-kT} = (Q^{(k)})(R^{(k)})^{-T}$. Now multiply by a vector, that is a load of zeros then a 1. We hence get, $\tilde{C}_k q_n^{(k)}$ and this converges to the smallest eigenvector.
Now we consider shifts in our algorithm. We now say,

$$A_k - s_k I = Q_k R_k$$

and then reverse,

$$A_{k+1} - R_k Q_k + s_k I.$$

The crux is the convergence is going to change. We have an error of,

$$\left(\frac{(\lambda - s_i)_{\min}}{(\lambda_i - s_i)_{\text{runner up}}} \right)$$

and note we don't invert the matrix.